Bounded Reachability Problems Are Decidable in FIFO Machines

Benedikt Bollig
LSV, ENS Paris-Saclay, CNRS, Université Paris-Saclay, France
benedikt.bollig@ens-paris-saclay.fr

Alain Finkel
LSV, ENS Paris-Saclay, CNRS, Université Paris-Saclay, France
alain.finkel@ens-paris-saclay.fr

Amrita Suresh
LSV, ENS Paris-Saclay, CNRS, Université Paris-Saclay, France
amrita.suresh@ens-paris-saclay.fr

Abstract

The undecidability of basic decision problems for general FIFO machines such as reachability and unboundedness is well-known. In this paper, we provide an underapproximation for the general model by considering only runs that are input-bounded (i.e. the sequence of messages sent through a particular channel belongs to a given bounded language). We prove, by reducing this model to a counter machine with restricted zero tests, that the rational-reachability problem (and by extension, control-state reachability, unboundedness, deadlock, etc.) is decidable. This class of machines subsumes input-letter-bounded machines, flat machines, linear FIFO nets, and monogeneous machines, for which some of these problems were already shown to be decidable. These theoretical results can form the foundations to build a tool to verify general FIFO machines based on the analysis of input-bounded machines.

2012 ACM Subject Classification Theory of computation

Keywords and phrases FIFO machines, reachability, underapproximation, counter machines

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2020.49

Related Version A full version of the paper is available at `https://hal.archives-ouvertes.fr/hal-02900813`.

1 Introduction

Context. Asynchronous distributed processes communicating using First In First Out (FIFO) channels are being widely used for distributed and concurrent programming, and more recently, for web service choreographies. Since systems of processes communicating through (at least two) one-directional FIFO channels, or equivalently, machines having a unique control-structure with a single FIFO channel (acting as a buffer) simulate Turing machines, most properties, such as unboundedness of a channel, are undecidable for such systems [31, 6, 30].

Reachability in FIFO machines. If one restricts to runs with B-bounded channels (the number of messages in every channel does not exceed B), then reachability becomes decidable for existentially-bounded and universally-bounded FIFO systems [20]. When limiting the number of phases, the bounded-context reachability problem is in 2-EXPTIME, even for recursive FIFO systems [27, 24]. For non-confluent topology, reachability is in EXPTIME for recursive FIFO systems with 1-bounded channels [24]. The notion of k-synchronous computations was introduced in [5]. Reachability under this restriction and checking k-synchronizability are both PSPACE-complete [22]. Reachability is in PTIME in half-duplex

systems [7] with two processes (moreover, the reachability set is recognizable and effectively computable), but the natural extension to three processes leads to undecidability. Lossy FIFO systems (where the channels can lose messages) [1, 16] have been shown to be well-structured and have a decidable (but non-elementary) reachability problem [9]. In [28, 2], uniform criteria for decidability of reachability and model-checking questions are established for communicating recursive systems whose restricted architecture or communication mechanism gives rise to behaviours of bounded tree-width.

Input-bounded FIFO machines. Many papers, starting in the 80s until today, have studied FIFO machines in which the input-language of a channel (i.e. the set of words that record the messages entering a channel) is included in the set $\mathit{Pref}(w_1^* w_2^* \dots w_n^*)$ of prefixes of a bounded language $w_1^* w_2^* \dots w_n^*$. We call this class of FIFO machines *input-bounded*.

If the *set of letters* that may enter a channel c is reduced to a unique letter a_c, then the input-language of c is included in a_c^* and this subclass trivially reduces to VASS and Petri nets [32]. Also note that, in general, the behaviour of those FIFO machines does not have bounded tree-width. *Monogeneous* FIFO nets [15, 30, 19] (input-languages of channels c are included in $LF(u_c v_c^*)$ where u_c, v_c are two words associated with c) and *linear* FIFO nets [17] (input-languages are included in $\mathit{Pref}(a_1^* a_2^* \dots a_n^*)$ where each a_i is a letter and $a_i \neq a_j$ iff $i \neq j$) both generalize Petri nets with still a decidable reachability problem. A variant of the reachability problem, the deadlock problem, is shown decidable for input-*letter*-bounded FIFO systems in [23] by reducing to reachability for VASS, but the extension to general input-bounded machines was left open.

Flat machines are another subclass of input-bounded machines in which the language of their control-graph, considered as a finite automaton, is a bounded language. For flat FIFO machines, control-state reachability is NP-complete [14]; this result has recently been extended to reachability, channel unboundedness, and other classical properties [18].

To the best of our knowledge, the decidability status of control-state reachability, reachability, deadlock, and termination was not known for input-bounded FIFO machines, which strictly include all the classes discussed above such as flat, input-letter-bounded, monogeneous, and linear FIFO machines (the last three types contain VASS and they are all incomparable). The unboundedness problem of input-bounded FIFO machines was shown decidable in [26] by using the well-structured concepts but with no extension to decidability of reachability.

Our contributions:

- We solve a problem that was left open in [23], the decidability of the reachability problem for input-bounded FIFO machines. We present a simulation of input-bounded FIFO machines by counter machines with restricted zero tests. The main idea is to associate a counter with each word in the bounded language, and to ensure that the counters are incremented and decremented in a way that corresponds to the FIFO order. Since we can have repeated letters, and ambiguities in the FIFO machine, we first need to construct a normal form of the FIFO machine. Furthermore, we ensure that for every run in the FIFO machine, we can construct an equivalent run in the counter machine and vice-versa.
- As we actually solve the general rational-reachability problem, we can deduce the decidability of other verification properties like control-state reachability, deadlock, unboundedness, and termination.
- We unify various definitions from the literature, survey the (not well-known) results, and generalize them.

- Following the bounded verification paradigm, applied to FIFO machines (for instance in [14, 18]), we open the way to a methodology that would apply existing results on input-bounded FIFO machines to general FIFO machines.

Plan. In Section 2, we present counter and FIFO machines, with the connection-deconnection protocol as an example. Section 3 contains the main result, which states the decidability of rational-reachability for FIFO machines restricted to input-bounded languages. Section 4 considers variants of the reachability problem such as unboundedness and termination. Finally, in Section 5, we mention further results, state some open problems, and discuss a possible theory of boundable FIFO machines. Missing proofs can be found in the long version of the paper, available at: `https://hal.archives-ouvertes.fr/hal-02900813`

2 Preliminaries

Words and Languages. Let A be a finite alphabet. As usual, A^* is the set of finite words over A, and A^+ the set of non-empty finite words. We let $|w|$ denote the length of $w \in A^*$. For the empty word ε, we have $|\varepsilon| = 0$. Given $a \in A$, let $|w|_a$ denote the number of occurrences of a in w. With this, we let $\mathit{Alph}(w) = \{a \in A \mid |w|_a \geq 1\}$. The concatenation of two words $u, v \in A^*$ is denoted by $u \cdot v$ or $u.v$ or simply uv. The sets of prefixes, suffixes, and infixes of $w \in A^*$ are denoted by $\mathit{Pref}(w)$, $\mathit{Suf}(w)$, and $\mathit{Infix}(w)$, resp. Note that $\{\varepsilon, w\} \subseteq \mathit{Pref}(w) \cap \mathit{Suf}(w) \cap \mathit{Infix}(w)$. For a set X, any mapping $f : A^* \to 2^X$ can be extended to $f : 2^{A^*} \to 2^X$ letting, for $L \subseteq A^*$, $f(L) = \bigcup_{w \in L} f(w)$. In particular, Alph, Pref, Suf, and Infix are extended in that way.

▶ **Definition 1** ([21]). *Let $w_1, \ldots, w_n \in A^+$ be non-empty words where $n \geq 1$. A* bounded language *over $(w_1, \ldots, w_n)$ is a language $L \subseteq w_1^* \ldots w_n^*$.*

We always assume that a bounded language L is given together with its tuple $(w_1, \ldots, w_n)$ and that $\mathit{Alph}(L) = \mathit{Alph}(w_1 \ldots w_n)$. We say that L is *distinct-letter* if $|w_1 \ldots w_n|_a \leq 1$ for all $a \in A$. If $|w_1| = \ldots = |w_n| = 1$, i.e. $w_1, \ldots, w_n \in A$, then L is a *letter-bounded language*. Let us remark that the set of bounded languages is closed under Pref and Suf.

Semi-Linear Sets. A *linear* set X (of dimension $d \geq 1$) is defined as a subset of $\mathbb{N}^d$ for which there exist a basis $\mathbf{b} \in \mathbb{N}^d$ and a finite set of periods $\{\mathbf{p}_1, \ldots, \mathbf{p}_m\} \subseteq \mathbb{N}^d$ such that $X = \{\mathbf{b} + \sum_{i=1}^m \lambda_i \mathbf{p}_i \mid \lambda_1, \ldots, \lambda_m \in \mathbb{N}\}$. A *semi-linear* set is defined as a finite union of linear sets.

Transition Systems. A *labeled transition system* is a quadruple $\mathcal{T} = (S, A, \to, \mathit{init})$ where S is the (potentially infinite) set of *configurations*[1], A is a finite alphabet, $\mathit{init} \in S$ is the *initial configuration*, and $\to \subseteq S \times A \times S$ is the *transition relation*.

For $s, s' \in S$, let $s \to s'$ if $s \xrightarrow{a} s'$ for some $a \in A$. For $w \in A^*$, we write $s \xrightarrow{w} s'$ if there is a w-labeled path from s to s'. Formally, $s \xrightarrow{\varepsilon} s'$ if $s = s'$, and $s \xrightarrow{aw} s'$ if there is $t \in S$ such that $s \xrightarrow{a} t$ and $t \xrightarrow{w} s'$. We let $\mathit{Traces}(\mathcal{T}) = \{w \in A^* \mid \mathit{init} \xrightarrow{w} s \text{ for some } s \in S\}$.

Given $w \in A^*$, we let $\mathit{Reach}_{\mathcal{T}}(w) = \{s \in S \mid \mathit{init} \xrightarrow{w} s\}$. Moreover, for $L \subseteq A^*$, $\mathit{Reach}_{\mathcal{T}}(L) = \bigcup_{w \in L} \mathit{Reach}_{\mathcal{T}}(w)$ is the set of configurations that are reachable via a word from L. Finally, the *reachability set* of $\mathcal{T}$ is defined as $\mathit{Reach}_{\mathcal{T}} = \mathit{Reach}_{\mathcal{T}}(A^*)$. We call $\mathcal{T}$ *finite* if $\mathit{Reach}_{\mathcal{T}}$ is finite (and this is the case if S is finite). Otherwise, $\mathcal{T}$ is called *infinite*.

[1] We say *configurations* rather than *states* to distinguish them from the *control states* used in FIFO and counter machines.

FIFO Machines. We consider FIFO machines having a sequential control graph rather than systems of communicating processes that are distributed systems. It is clear that, given a distributed system, one may compute the Cartesian product of all processes to obtain a FIFO machine (the converse is not always true).

▶ **Definition 2.** *A* FIFO machine *is a tuple* $M = (Q, Ch, \Sigma, T, q_0)$ *where* Q *is a finite set of* control states*,* $q_0 \in Q$ *is an* initial control state*, and* Ch *is a finite set of* channels*. Moreover,* Σ *is a finite* message alphabet*. It is partitioned into* $\Sigma = \biguplus_{c \in Ch} \Sigma_c$ *where* Σ_c *contains the messages that can be sent through channel* c*. Finally,* $T \subseteq Q \times A_M \times Q$ *is a* transition relation *where* $A_M = \{\langle c!a \rangle \mid c \in Ch \text{ and } a \in \Sigma_c\} \cup \{\langle c?a \rangle \mid c \in Ch \text{ and } a \in \Sigma_c\}$ *is the set of send and receive actions.*

▶ **Example 3** (Connection-Deconnection Protocol)**.** A model for the (simplified) connection-deconnection protocol, CDP, between two processes is described as follows (see Figure 1): We model the protocol with two automata (representing the two processes) and two (infinite) channels. The first processes (on the left) can open a session (this is denoted by sending the message "a" through channel c_1 to the other process). Once a session is open, the first process can close it (by sending message "b" to the other process), or on the demand of the second process (if it receives the message "e"). This protocol has been studied in [25].

In the example, it is natural to have two separate processes. However, following Definition 2, we formalize this in terms of the Cartesian product of the two processes. That is, the CDP is modeled as the FIFO machine $M = (Q, Ch, \Sigma, T, q_0)$ where $Q = \{0, 1\} \times \{0, 1\}$ (the Cartesian product of the local state spaces) with initial state $q_0 = (0, 0)$, $Ch = \{c_1, c_2\}$, $\Sigma = \Sigma_{c_1} \uplus \Sigma_{c_2}$ with $\Sigma_{c_1} = \{a, b\}$ and $\Sigma_{c_2} = \{e\}$. Moreover, the transition relation T contains, amongst others, $((0, 0), \langle c_1!a \rangle, (1, 0))$ and $((1, 0), \langle c_1?a \rangle, (1, 1))$. ⌟

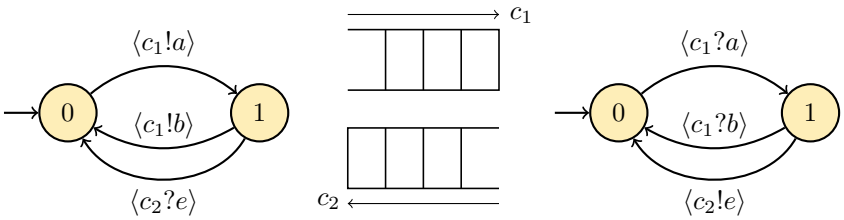

Figure 1 The model of the connection-deconnection protocol.

A FIFO machine $M = (Q, Ch, \Sigma, T, q_0)$ induces a (potentially infinite) transition system $\mathcal{T}_M = (S_M, A_M, \to_M, init_M)$. Its set of configurations is $S_M = Q \times \prod_{c \in Ch} \Sigma_c^*$. In $(q, \mathbf{w}) \in S_M$, the first component q denotes the current control state and $\mathbf{w} = (\mathbf{w}_c)_{c \in Ch}$ determines the contents $\mathbf{w}_c \in \Sigma_c^*$ for every channel $c \in Ch$. The initial configuration is $init_M = (q_0, \boldsymbol{\varepsilon})$ where $\boldsymbol{\varepsilon} = (\varepsilon, \ldots, \varepsilon)$, i.e., every channel is empty. The transitions are given as follows:

- $(q, \mathbf{w}) \xrightarrow{\langle c!a \rangle}_M (q', \mathbf{w}')$ if $(q, \langle c!a \rangle, q') \in T$, $\mathbf{w}'_c = \mathbf{w}_c \cdot a$, and $\mathbf{w}'_d = \mathbf{w}_d$ for all $d \in Ch \setminus \{c\}$;
- $(q, \mathbf{w}) \xrightarrow{\langle c?a \rangle}_M (q', \mathbf{w}')$ if $(q, \langle c?a \rangle, q') \in T$, $\mathbf{w}_c = a \cdot \mathbf{w}'_c$, and $\mathbf{w}'_d = \mathbf{w}_d$ for all $d \in Ch \setminus \{c\}$.

The index M may be omitted whenever M is clear from the context.

The *reachability set* of M is defined as the reachability set of $\mathcal{T}_M$, i.e., $Reach_M = Reach_{\mathcal{T}_M}$ and, for $L \subseteq A_M^*$, $Reach_M(L) = Reach_{\mathcal{T}_M}(L)$. Moreover, we let $Traces(M) = Traces(\mathcal{T}_M)$.

▶ **Example 4.** An example run of the FIFO machine M from Example 3 and Figure 1 is $((0, 0), (\varepsilon, \varepsilon)) \xrightarrow{\langle c_1!a \rangle} ((1, 0), (a, \varepsilon)) \xrightarrow{\langle c_1?a \rangle} ((1, 1), (\varepsilon, \varepsilon)) \xrightarrow{\langle c_2!e \rangle} ((1, 0), (\varepsilon, e))$. As for the reachability set, we have, e.g., $((1, 1), ((ba)^*, \varepsilon))) \subseteq Reach_M$ and $((0, 0), (b(ab)^*, e)) \subseteq Reach_M$.

Let us remark that CDP is not half-duplex because there are reachable configurations with both channels non-empty, e.g., $((0,0),(b,e))$; moreover, it is neither monogeneous, nor linear, nor input-letter-bounded. ⌟

Counter Machines. We next recall the notion of counter machines, where multiple counters can take non-negative integer values, be incremented and decremented, and be tested for zero (though in a restricted fashion).

▶ **Definition 5.** *A* counter machine *(with zero tests) is a tuple* $\mathcal{C} = (Q, Cnt, T, q_0)$*. Like in a FIFO machine,* Q *is the finite set of* control states *and* $q_0 \in Q$ *is the* initial control state*. Moreover,* Cnt *is a finite set of* counters *and* $T \subseteq Q \times A_{\mathcal{C}} \times Q$ *is the transition relation where* $A_{\mathcal{C}} = \{\mathsf{inc}(x), \mathsf{dec}(x) \mid x \in Cnt\} \times 2^{Cnt}$*.*

The counter machine $\mathcal{C}$ induces a transition system $\mathcal{T}_{\mathcal{C}} = (S_{\mathcal{C}}, A_{\mathcal{C}}, \to_{\mathcal{C}}, init_{\mathcal{C}})$ with set of configurations $S_{\mathcal{C}} = Q \times \mathbb{N}^{Cnt}$. In $(q, \mathbf{v}) \in S_{\mathcal{C}}$, q is the current control state and $\mathbf{v} = (\mathbf{v}_x)_{x \in Cnt}$ represents the counter values. The initial configuration is $init_{\mathcal{C}} = (q_0, \mathbf{0})$ where $\mathbf{0}$ maps all counters to 0. For $op \in \{\mathsf{inc}, \mathsf{dec}\}$, $x \in Cnt$, and $Z \subseteq Cnt$ (the counters tested for zero), there is a transition $(q, \mathbf{v}) \xrightarrow{(op(x),Z)}_{\mathcal{C}} (q', \mathbf{v}')$ if $(q, (op(x), Z), q') \in T$, $\mathbf{v}_y = 0$ for all $y \in Z$ (applies the zero tests), $\mathbf{v}'_x = \mathbf{v}_x + 1$ if $op = \mathsf{inc}$ and $\mathbf{v}'_x = \mathbf{v}_x - 1$ if $op = \mathsf{dec}$, and $\mathbf{v}'_y = \mathbf{v}_y$ for all $y \in Cnt \setminus \{x\}$.

The *reachability set* of $\mathcal{C}$ is defined as $Reach_{\mathcal{C}} = Reach_{\mathcal{T}_{\mathcal{C}}}$. For $L \subseteq A_{\mathcal{C}}^*$, we also let $Reach_{\mathcal{C}}(L) = Reach_{\mathcal{T}_{\mathcal{C}}}(L)$. Moreover, $Traces(\mathcal{C}) = Traces(\mathcal{T}_{\mathcal{C}})$. To get decidability of reachability in counter machines, we impose the restriction that, once a counter has been tested for zero, it cannot be incremented or decremented anymore. This is clearly an extension of VASS. To define this, let $L_{\mathcal{C}}^{\mathsf{zero}}$ be the set of words $(op_1(x_1), Z_1) \dots (op_n(x_n), Z_n) \in A_{\mathcal{C}}^*$ such that, for every two positions $1 \le i \le j \le n$, we have $x_j \notin Z_i$.

▶ **Theorem 6.** *The following problem is decidable (though inherently non-elementary): Given a counter machine* $\mathcal{C} = (Q, Cnt, T, q_0)$*, a regular language* $L \subseteq A_{\mathcal{C}}^*$*, a control state* $q \in Q$*, and a semi-linear set* $V \subseteq \mathbb{N}^{Cnt}$*, do we have* $(q, \mathbf{v}) \in Reach_{\mathcal{C}}(L_{\mathcal{C}}^{\mathsf{zero}} \cap L)$ *for some* $\mathbf{v} \in V$*?*

Proof sketch. Reachability in presence of a semi-linear target set and restricted zero tests straightforwardly reduces to configuration-reachability in counter machines without zero tests (i.e., VASS and Petri nets). The latter is decidable [29], though inherently non-elementary [11]. First, zero tests are postponed to the very end of an execution and, to this aim, stored in the control-state. Second, to check whether a counter valuation is contained in V, we can branch, whenever we are in the given control-state q, into a new component that decrements counters accordingly and eventually checks whether they are all zero. ◀

3 The Input-Bounded Rational-Reachability Problem

It is very well known that the following reachability problem is undecidable: Given a FIFO machine $M = (Q, Ch, \Sigma, T, q_0)$, a configuration $(q, \mathbf{w}) \in S_M$, and a regular language $L \subseteq A_M^*$, do we have $(q, \mathbf{w}) \in Reach_M(L)$? Of course, the problem is already undecidable when we impose $L = A_M^*$. Motivated by this negative result, we are looking for language classes $\mathfrak{C}$ that render the problem decidable under the restriction that $L \in \mathfrak{C}$.

We say that a FIFO machine $M = (Q, Ch, \Sigma, T, q_0)$ has a *bounded reachability set* if there is a tuple $(L_c)_{c \in Ch}$ of regular bounded languages $L_c \subseteq \Sigma_c^*$ such that, for all $(q, \mathbf{w}) \in Reach_M$, we have $\mathbf{w} \in \prod_{c \in Ch} L_c$. We observe that restricting the reachability set to be bounded is not sufficient to obtain a decidable reachability problem. We show this by simulating any two counter Minsky machine by a FIFO machine with fixed languages L_c.

▶ **Theorem 7.** *The reachability problem is undecidable for FIFO machines with a (given) bounded reachability set.*

We therefore consider a different restriction to obtain decidability. For a given FIFO machine $M = (Q, Ch, \Sigma, T, q_0)$, we are interested in $Reach_M(L)$ where $L \subseteq A_M^*$ is *input-bounded* in the following sense: For every channel c, the sequence of messages that are sent through channel c is from a given regular bounded language $L_c \subseteq \Sigma_c^*$.

Let us be more formal. For $c \in Ch$, we let $proj_{c!} : A_M^* \to \Sigma_c^*$ be the homomorphism defined by $proj_{c!}(\langle c!a\rangle) = a$ for all $a \in \Sigma_c$, and $proj_{c!}(\beta) = \varepsilon$ if $\beta \in A_M$ is not of the form $\langle c!a\rangle$ for some $a \in \Sigma_c$. We define $proj_{c?} : A_M^* \to \Sigma_c^*$ accordingly.

With this, given a tuple $\mathscr{L} = (L_c)_{c\in Ch}$ of bounded languages $L_c \subseteq \Sigma_c^*$, we set $\mathscr{L}_! = \{\sigma \in A_M^* \mid proj_{c!}(\sigma) \in L_c \text{ for all } c \in Ch\}$ and $\mathscr{L}_? = \{\sigma \in A_M^* \mid proj_{c?}(\sigma) \in L_c \text{ for all } c \in Ch\}$. We observe that, if all L_c are regular, then so are $\mathscr{L}_!$ and $\mathscr{L}_?$.

▶ **Definition 8.** *The* input-bounded (IB) reachability problem *asks whether a given configuration* $(q, \mathbf{w})$ *is reachable along a sequence of actions from* $\mathscr{L}_!$*, i.e., whether* $(q, \mathbf{w}) \in Reach_M(\mathscr{L}_!)$.

Note that, if $(q_0, \boldsymbol{\varepsilon}) \xrightarrow{\sigma}_M (q, \mathbf{w})$ and $\sigma \in \mathscr{L}_!$, then we also have $\sigma \in Pref(\mathscr{L}_?)$ due to the FIFO policy. Thus, $Reach_M(\mathscr{L}_!) = Reach_M(\mathscr{L}_! \cap Pref(\mathscr{L}_?))$ so that we can restrict to action sequences from $\mathscr{L}_! \cap Pref(\mathscr{L}_?)$. We will call $\mathscr{L}_! \cap Pref(\mathscr{L}_?)$ the set of *valid* words.

▶ **Example 9.** Let us come back to the protocol CDP M from Example 3 and Figure 1, which is neither monogeneous nor linear nor flat. Since the "input-languages" of the two channels (i.e. the languages of words that record the messages entering a channel) contain $\{a, ab\}^*$ and e^*, resp., and since $\{a, ab\}^*$ is not a bounded language, we have $Traces(M) \not\subseteq \mathscr{L}_!$ for every pair of bounded languages $\mathscr{L}$. In other words, M is not input-bounded. However, when we look at the reachability set obtained by considering the tuple of bounded languages $\mathscr{L} = (L_{c_1}, L_{c_2})$ where $L_{c_1} = (ab)^*(a + \varepsilon)(ab)^*$ is a bounded language over (ab, a, ab), and $L_{c_2} = e^*$ is a bounded language over (e), we still obtain the entire reachability set. That is, we have $Reach_M = Reach_M(\mathscr{L}_!)$. Hence, even though the input-languages of the system are not all bounded, we can still compute the reachability set by restricting our exploration to a tuple of (regular) bounded languages $\mathscr{L}$. ⌟

Actually, instead of reachability of a single configuration as stated in Definition 8, we study a more general problem, called the *input-bounded rational-reachability problem*. It asks whether a configuration $(q, \mathbf{w})$ is reachable for some channel contents $\mathbf{w}$ from a given *rational* relation. So let us define rational relations.

Rational and Recognizable Relations. Consider a relation $\mathscr{R} \subseteq \prod_{c\in Ch} \Sigma_c^*$. We say that $\mathscr{R}$ is *rational* if there is a regular word language $R \subseteq \Theta^*$ over the alphabet $\Theta = \prod_{c\in Ch}(\Sigma_c \cup \{\varepsilon\})$ such that $\mathscr{R} = \{(\mathbf{a}_c^1 \cdot \ldots \cdot \mathbf{a}_c^n)_{c\in Ch} \mid \mathbf{a}^1 \ldots \mathbf{a}^n \in R \text{ with } n \in \mathbb{N} \text{ and } \mathbf{a}^i = (\mathbf{a}_c^i)_{c\in Ch} \in \Theta \text{ for } i \in \{1, \ldots, n\}\}$. Here, $\mathbf{a}_c^1 \cdot \ldots \cdot \mathbf{a}_c^n \in \Sigma_c^*$ is the concatenation of all $\mathbf{a}_c^i \in \Sigma_c \cup \{\varepsilon\}$ while ignoring the neutral element ε. For example, in the presence of two channels, $\mathscr{R} = \{(a^m, b^n) \mid m \geq n\}$ is a rational relation, witnessed by $R = ((a, b) + (a, \varepsilon))^*$. In the following, we will always assume that a rational relation is given in terms of a finite automaton for the underlying regular language R.

A relation $\mathscr{R} \subseteq \prod_{c\in Ch} \Sigma_c^*$ is called *recognizable* if it is the finite union of relations of the form $\prod_{c\in Ch} R_c$ where all $R_c \subseteq \Sigma_c^*$ are regular languages. Note that every recognizable relation is rational while the converse is, in general, false.

We define the *Parikh image* of a relation $\mathscr{R} \subseteq \prod_{c \in Ch} \Sigma_c^*$ as $\mathrm{Parikh}(\mathscr{R}) = \{(\pi_a)_{a \in \Sigma} \in \mathbb{N}^\Sigma \mid \exists \mathbf{w} = (\mathbf{w}_c)_{c \in Ch} \in \mathscr{R} : \pi_a = |\mathbf{w}_c|_a \text{ for all } c \in Ch \text{ and } a \in \Sigma_c\}$. It is well known that, if $\mathscr{R}$ is rational, then $\mathrm{Parikh}(\mathscr{R})$ is semi-linear.

For more background on rational relations and their subclasses, we refer to [4, 10].

The IB Rational-Reachability Problem. We are now prepared to define the input-bounded (IB) rational-reachability problem and to state its decidability:

▶ **Definition 10.** *The* IB rational-reachability problem *is defined as follows: Given a FIFO machine $M = (Q, Ch, \Sigma, T, q_0)$, a tuple $\mathscr{L} = (L_c)_{c \in Ch}$ of non-empty regular bounded languages $L_c \subseteq \Sigma_c^*$ (each given in terms of a finite automaton), a control state $q \in Q$, and a rational relation $\mathscr{R} \subseteq \prod_{c \in Ch} \Sigma_c^*$. Do we have $(q, \mathbf{w}) \in Reach_M(\mathscr{L}_!)$ for some $\mathbf{w} \in \mathscr{R}$?*

▶ **Theorem 11.** *IB rational-reachability is decidable for FIFO machines.*

The remainder of this section is devoted to the proof of Theorem 11.

Let $M = (Q, Ch, \Sigma, T, q_0)$ and let $\mathscr{L} = (L_c)_{c \in Ch}$ be a tuple of non-empty regular bounded languages $L_c \subseteq \Sigma_c^*$ over $(w_{c,1}, \ldots, w_{c,n_c})$. We proceed by reduction to counter machines. The rough idea is to represent the contents of channel c in terms of several counters, one for every component $w_{c,i}$. To have a faithful simulation, we rely on a normal form of M and its bounded languages, which can be achieved at the expense of an exponential blow-up of the FIFO machine.

▶ **Definition 12.** *We say that M and $\mathscr{L}$ are in* normal form *if the following hold:*

1. *For all $c \in Ch$, $\Sigma_c \subseteq Alph(L_c)$ and L_c is distinct-letter.*
2. *We have $Traces((Q, A_M, T, q_0)) \subseteq Pref(\mathcal{V})$ where $\mathcal{V} = \mathscr{L}_! \cap Pref(\mathscr{L}_?)$. Note that (Q, A_M, T, q_0) is the finite transition system induced by the control graph of M.*

Given a FIFO machine $\hat{M} = (\hat{Q}, Ch, \hat{\Sigma}, \hat{T}, \hat{q}_0)$ and the tuple $\hat{\mathscr{L}} = (\hat{L}_c)_{c \in Ch}$ of non-empty regular bounded languages $\hat{L}_c \subseteq \hat{\Sigma}_c^*$, we now construct $M = (Q, Ch, \Sigma, T, q_0)$ and $\mathscr{L} = (L_c)_{c \in Ch}$ in normal form such that a reachability query in the former can be transformed into a reachability query in the latter (made precise in Lemma 15 below).

Distinct-Letter Property. Consider the bounded language $\hat{L}_c$ over $(\hat{w}_{c,1}, \ldots, \hat{w}_{c,n_c})$. For $i \in \{1, \ldots, n_c\}$, let $m_i = |\hat{w}_{c,1}| + \ldots + |\hat{w}_{c,i}|$ be the number of letters in the first i words. Moreover, $m = m_{n_c}$. Let Σ_c denote the alphabet $\{a_1^c, \ldots, a_m^c\}$. It contains the "distinct" letters for the bounded language L_c over $(w_{c,1}, \ldots, w_{c,n_c})$, where we let $w_{c,1} = a_1^c \ldots a_{m_1}^c$ and $w_{c,i} = a_{m_{i-1}+1}^c \ldots a_{m_i}^c$ for $i \geq 2$. In other words, the letters in $(w_{c,1}, \ldots, w_{c,n_c})$ are numbered consecutively. In order to obtain the language L_c, we first consider the homomorphism $h_c : \Sigma_c^* \to \hat{\Sigma}_c^*$ where $h_c(a_i^c)$ is the i-th letter in the word $\hat{w}_{c,1} \ldots \hat{w}_{c,n_c}$. We obtain L_c as $h_c^{-1}(\hat{L}_c) \cap (w_{c,1})^* \ldots (w_{c,n_c})^*$, hence preserving regularity and boundedness. We then remove those words from $(w_{c,1}, \ldots, w_{c,n_c})$ (and their letters from Σ_c) whose letters do not occur in L_c. We have $\Sigma_c \subseteq Alph(L_c)$.

▶ **Example 13.** For example, suppose we have one channel c and $\hat{L}_c = (ab)^*bb^*$ over (ab, b). We determine the language L_c over (a_1a_2, a_3) (omitting the superscript c in the letters). The homomorphism $h_c : \{a_1, a_2, a_3\}^* \to \{a, b\}^*$ is given by $h_c(a_1) = a$ and $h_c(a_2) = h_c(a_3) = b$. We have $h_c^{-1}(\hat{L}_c) = (a_1(a_2 + a_3))^*(a_2 + a_3)(a_2 + a_3)^*$, which we intersect with $(a_1a_2)^*a_3^*$. We thus get the regular bounded language $L_c = (a_1a_2)^*a_3a_3^*$ over (a_1a_2, a_3). All letters from $\{a_1, a_2, a_3\}$ occur in L_c so that we are done.

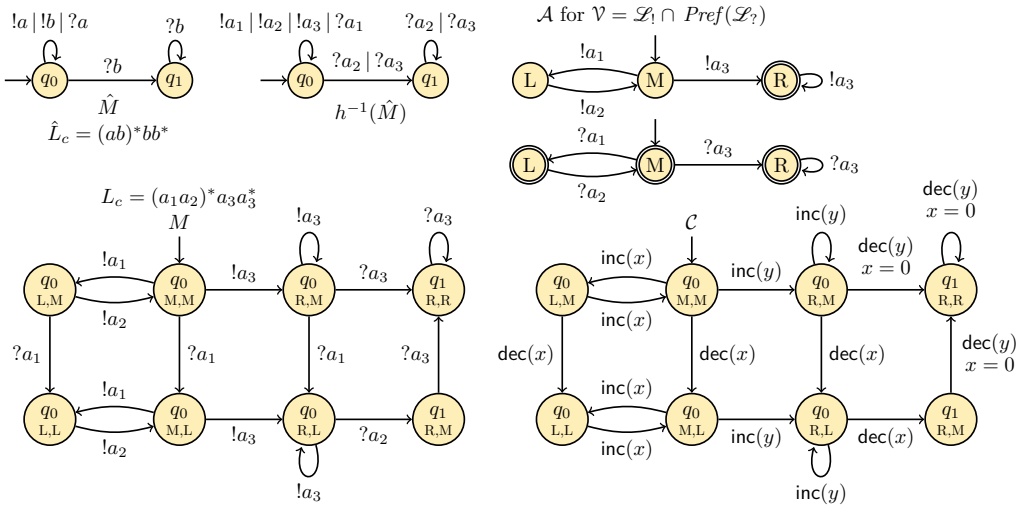

Figure 2 For a FIFO machine $\hat{M}$ with a single channel c and the bounded language $\hat{L}_c = (ab)^*bb^*$ over (ab, b) (top leftmost), we construct a FIFO machine M (bottom left), together with $L_c = (a_1a_2)^*a_3a_3^*$, in normal form as the product of $h^{-1}(\hat{M})$ and an automaton for $\mathcal{V}$ (top right). From M, we then obtain the counter machine $\mathcal{C}$ (bottom right).

Trace Property. In the next step, we build the FIFO machine $M = (Q, Ch, \Sigma, T, q_0)$ such that $Traces((Q, A_M, T, q_0)) \subseteq Pref(\mathcal{V})$ with $\mathcal{V} = \mathscr{L}_! \cap Pref(\mathscr{L}_?)$. First, to take care of the homomorphisms h_c, we define the transition relation $h^{-1}(\hat{T}) = \{(q, \langle c!e\rangle, q') \mid (q, \langle c!a\rangle, q') \in \hat{T}$ and $e \in h_c^{-1}(a)\} \cup \{(q, \langle c?e\rangle, q') \mid (q, \langle c?a\rangle, q') \in \hat{T}$ and $e \in h_c^{-1}(a)\}$. Thus, the set of actions of M will be $A_M = \{\langle c!e\rangle \mid c \in Ch$ and $e \in \Sigma_c\} \cup \{\langle c?e\rangle \mid c \in Ch$ and $e \in \Sigma_c\}$.

To continue our above example, a transition $(q, \langle c!b\rangle, q')$ would be replaced with the two transitions $(q, \langle c!a_2\rangle, q')$ and $(q, \langle c!a_3\rangle, q')$, and similarly for $(q, \langle c?b\rangle, q')$.

To guarantee trace inclusion in $Pref(\mathcal{V})$, we will consider a deterministic (not necessarily complete) finite automaton $\mathcal{A} = (Q_\mathcal{A}, A_M, T_\mathcal{A}, q^0_\mathcal{A}, F_\mathcal{A})$, with set of final states $F_\mathcal{A} \subseteq Q_\mathcal{A}$, whose language is $L(\mathcal{A}) = \mathcal{V}$ and where, from every state, a final state is reachable in the finite graph $(Q_\mathcal{A}, T_\mathcal{A})$. With this, we define M as the product of the FIFO machine $h^{-1}(\hat{M}) = (\hat{Q}, Ch, \Sigma, h^{-1}(\hat{T}), \hat{q}_0)$ and $\mathcal{A}$ in the expected manner. In particular, the set of control states of M is $\hat{Q} \times Q_\mathcal{A}$, and its initial state is the pair $(\hat{q}_0, q^0_\mathcal{A})$.

▶ **Example 14.** Figure 2 illustrates the result of the normalization procedure for a FIFO machine $\hat{M}$ with one single channel c (which is therefore omitted) and its bounded language $\hat{L}_c = (ab)^*bb^*$ over (ab, b). Recall from Example 13 that the corresponding homomorphism h_c maps a_1 to a and both a_2 and a_3 to b, and that we obtain $L_c = (a_1a_2)^*a_3a_3^*$. Moreover, M is the product of $h^{-1}(\hat{M})$ (depicted in the top center) and a finite automaton $\mathcal{A}$ for $\mathcal{V} = \mathscr{L}_! \cap Pref(\mathscr{L}_?)$ (obtained as the shuffle of the two finite automata on the top right). The state names in M reflect the states of $\hat{M}$ and $\mathcal{A}$ they originate from. We depict only accessible states of M from which we can still complete the word read so far to a word in $\mathcal{V}$. For example, $(q_1, \mathrm{M}, \mathrm{L})$ and $(q_1, \mathrm{L}, \mathrm{R})$ would no longer allow us to reach the final state R of the $\mathscr{L}_!$-component. ⌟

Now suppose we are given a reachability query for $\hat{M}$ in terms of $\hat{q} \in \hat{Q}$ and a rational relation $\hat{\mathscr{R}} \subseteq \prod_{c \in Ch} \hat{\Sigma}_c^*$. The lemma below shows how to reduce it to a reachability query in M. Here, for $\mathbf{w} = (\mathbf{w}_c)_{c \in Ch} \in \prod_{c \in Ch} \Sigma_c^*$, we define $h(\mathbf{w}) = (h_c(\mathbf{w}_c))_{c \in Ch} \in \prod_{c \in Ch} \hat{\Sigma}_c^*$. Note that $h^{-1}(\hat{\mathscr{R}})$ is rational.

▶ **Lemma 15.** *We have* $(\hat{q}, \hat{\mathbf{w}}) \in Reach_{\hat{M}}(\hat{\mathscr{L}}_!)$ *for some* $\hat{\mathbf{w}} \in \hat{\mathscr{R}}$ *iff* $((\hat{q}, q_{\mathcal{A}}), \mathbf{w}) \in Reach_M(\mathscr{L}_!)$ *for some* $q_{\mathcal{A}} \in Q_{\mathcal{A}}$ *and* $\mathbf{w} \in h^{-1}(\hat{\mathscr{R}})$.

Reduction of Normal Form to Counter Machine

Henceforth, we suppose that $M = (Q, Ch, \Sigma, T, q_0)$ and $\mathscr{L} = (L_c)_{c \in Ch}$ are in normal form, where L_c is a bounded language over $(w_{c,1}, \ldots, w_{c,n_c})$. In particular, for every letter $a \in \Sigma_c$, there is a unique index $i \in \{1, \ldots, n_c\}$ such that $a \in \Sigma_{c,i}$ where $\Sigma_{c,i} = Alph(w_{c,i})$. We denote this index i by i_a.

We build a counter machine $\mathcal{C}$ such that the IB rational-reachability problem for M can be solved by answering a reachability query in $\mathcal{C}$, using Theorem 6. Each run in $\mathcal{C}$ will simulate a run in M. In particular, we want a configuration of $\mathcal{C}$ to allow us to draw conclusions about the simulated configuration in M. The difficulty here is that counter values are just natural numbers and a priori store less information than channel contents with their messages. To overcome this, the idea is to represent each word $w_{c,i}$ of a tuple $(w_{c,1}, \ldots, w_{c,n_c})$ as a counter $x_{(c,i)}$. Since the set of possible action sequences is "guided" by a bounded language, we can replace send actions with increments and receive actions with decrements. More precisely, $\langle c!a \rangle$ becomes $(\mathsf{inc}(x_{(c,i_a)}), \emptyset)$, thus incrementing the counter associated with the unique word $w_{c,i}$ in which a occurs. Similarly, $\langle c?a \rangle$ translates to $(\mathsf{dec}(x_{(c,i_a)}), Z)$ (for suitable Z).

This alone does not put us in a position yet where, from a counter valuation, we can infer a unique channel contents. However, when we additionally keep track of the last messages that have been sent for each channel, we can reconstruct a unique channel contents.

There is one more thing to consider here. While the counters $x_{(c,i)}$ for a given channel c are kind of independent, the FIFO policy would not allow us to receive a letter from $w_{c,j}$ while a letter from $w_{c,i}$ with $i < j$ is in transit. Translated to the counter setting, this means that performing $\mathsf{dec}(x_{(c,j)})$ should require all counters $x_{(c,i)}$ with $i < j$ to be 0, so this is where zero tests come into play. As the L_c are bounded languages and thanks to the normal form, however, a counter that has been tested for zero does not need to be modified anymore.

We can directly implement these ideas formally and define $\mathcal{C} = (Q, Cnt, T', q_0)$ as follows (note that Q and q_0 remain unchanged):

- The set of counters is $Cnt = \{x_{(c,i)} \mid c \in Ch \text{ and } i \in \{1, \ldots, n_c\}\}$.
- For every $(q, \langle c!a \rangle, q') \in T$, we have $(q, (\mathsf{inc}(x_{(c,i_a)}), \emptyset), q') \in T'$.
- For every $(q, \langle c?a \rangle, q') \in T$, we have $(q, (\mathsf{dec}(x_{(c,i_a)}), Z), q') \in T'$ where the set of counters to be tested for zero is $Z = \{x_{(c,j)} \mid j < i_a\}$.

▶ **Example 16.** Figure 2 illustrates the construction of $\mathcal{C}$ from a FIFO machine M in normal form (cf. Example 14). Recall that we have one channel c and the bounded language $L_c = (a_1 a_2)^* a_3 a_3^*$ over $(a_1 a_2\,, a_3)$. Thus, $\mathcal{C}$ will have two counters, say x for $a_1 a_2$ and y for a_3. Note that performing $\mathsf{dec}(y)$ indeed comes with a test of x for zero.

Let us first observe that it is actually important that the FIFO machine satisfies the trace property. Suppose that, rather than from M, we constructed the counter machine directly from $h^{-1}(\hat{M})$. Then, configuration $(q_1, (1, 0))$ would be reachable in the counter machine via $\mathsf{inc}(x)\mathsf{inc}(x)\mathsf{dec}(x)$, which arises from $\langle c!a_1 \rangle \langle c!a_2 \rangle \langle c?a_2 \rangle$. However the only corresponding trace from $Pref(\mathcal{V})$ is $\langle c!a_1 \rangle \langle c!a_2 \rangle \langle c?a_1 \rangle$, which in the FIFO machine $h^{-1}(\hat{M})$ leads to q_0.

So consider M and its counter machine $\mathcal{C}$. A channel contents $\mathbf{w} \in \Sigma_c^*$ (here, we have one channel) has a natural counter analogue $\langle\!\langle \mathbf{w} \rangle\!\rangle = (|\mathbf{w}|_{a_1} + |\mathbf{w}|_{a_2}, |\mathbf{w}|_{a_3})$. In fact, if $(\overline{q}, \mathbf{w})$ is reachable in M, then following the corresponding transitions in $\mathcal{C}$ will lead us to $(\overline{q}, \langle\!\langle \mathbf{w} \rangle\!\rangle)$. For example, $((q_0, \mathrm{R}, \mathrm{L}), a_2 a_3 a_3)$ is reachable in M along the trace $\langle c!a_1 \rangle \langle c!a_2 \rangle \langle c?a_1 \rangle \langle c!a_3 \rangle \langle c!a_3 \rangle$, and so is $((q_0, \mathrm{R}, \mathrm{L}), (1, 2))$ in $\mathcal{C}$ along $\mathsf{inc}(x)\mathsf{inc}(x)\mathsf{dec}(x)\mathsf{inc}(y)\mathsf{inc}(y)$ (all zero tests are empty).

But how about the converse? In general, one may associate with a counter valuation such as $(4,0)$ several channel contents. Actually, both $a_1a_2a_1a_2$ and $a_2a_1a_2a_1$ seem suitable. However, if we know the most recent message that has been sent, say a_1, then this leaves only one option, namely $a_2a_1a_2a_1$. In this way, we can associate with each counter valuation $\mathbf{v}$ and message $a_i \in \Sigma_c$ a unique (if it exists at all) possible channel contents $[\![\mathbf{v}]\!]_{a_i}$. Suppose that τ is a trace in $\mathcal{C}$ arising from a trace σ in M whose last sent message is a_i. If $(\overline{q}, \mathbf{v})$ is reachable in $\mathcal{C}$ via τ, then $(\overline{q}, [\![\mathbf{v}]\!]_{a_i})$ is reachable in M via σ. For example, $\tau = \mathsf{inc}(x)\mathsf{inc}(x)\mathsf{dec}(x)$ allows us to go to configuration $((q_0, \mathrm{M}, \mathrm{L}), (1,0))$. It arises from $\sigma = \langle c!a_1\rangle\langle c!a_2\rangle\langle c?a_1\rangle \in \mathit{Pref}(\mathcal{V})$, whose last sent message is a_2. We have $[\![(1,0)]\!]_{a_2} = a_2$. Indeed, σ leads to $((q_0, \mathrm{M}, \mathrm{L}), a_2)$. ⌟

Relation between FIFO Machine and Counter Machine

Recall that the FIFO machine $M = (Q, \mathit{Ch}, \Sigma, T, q_0)$ and $\mathscr{L} = (L_c)_{c\in\mathit{Ch}}$ are in normal form, where L_c is a bounded language over $(w_{c,1}, \ldots, w_{c,n_c})$. Let $\mathcal{C} = (Q, \mathit{Cnt}, T', q_0)$ be the associated counter machine. We will now formalize the tight forth-and-back correspondence that allows us to solve reachability queries in M in terms of reachability queries in $\mathcal{C}$.

We start with a simple observation concerning the traces of M and $\mathcal{C}$.

▶ **Lemma 17.** *We have* $\mathit{Traces}(M) \subseteq \mathit{Pref}(\mathcal{V})$ *and* $\mathit{Traces}(\mathcal{C}) \subseteq L_{\mathcal{C}}^{\mathsf{zero}}$.

With every channel contents $\mathbf{w} \in \prod_{c\in\mathit{Ch}} \Sigma_c^*$ of the FIFO machine M, we associate a counter valuation $\langle\!\langle \mathbf{w} \rangle\!\rangle = \mathbf{v} \in \mathbb{N}^{\mathit{Cnt}}$ where, for each counter $x_{(c,i)}$, we let $\mathbf{v}_{x_{(c,i)}} = \sum_{a\in\Sigma_{c,i}} |\mathbf{w}_c|_a$. Furthermore, abusing notation, we define a homomorphism $\langle\!\langle\, . \,\rangle\!\rangle : A_M^* \to A_{\mathcal{C}}^*$ which maps a sequence of actions of M to a sequence of actions of $\mathcal{C}$. It is defined by $\langle\!\langle \langle c!a\rangle \rangle\!\rangle = (\mathsf{inc}(x_{(c,i_a)}), \emptyset)$ and $\langle\!\langle \langle c?a\rangle \rangle\!\rangle = (\mathsf{dec}(x_{(c,i_a)}), Z)$ where $Z = \{x_{(c,j)} \mid j < i_a\}$.

Conversely, we will associate, with counter values and traces of $\mathcal{C}$ the corresponding objects in the FIFO machine. Because of the inherent ambiguity, this is, however, less straightforward. First, we define a partial mapping $[\![\, . \,]\!] : A_{\mathcal{C}}^* \to A_M^*$ (that is not a homomorphism). For $\tau \in A_{\mathcal{C}}^*$, we let $[\![\tau]\!]$ be the unique (if it exists) word $\sigma \in \mathit{Pref}(\mathcal{V})$ such that $\langle\!\langle \sigma \rangle\!\rangle = \tau$.

Next, we associate with a counter valuation a corresponding channel contents. As explained above, there is no unique choice unless we make an assumption on the last messages that have been sent. For $c \in \mathit{Ch}$, we set $\Sigma_c^\perp = \Sigma_c \uplus \{\perp\}$. Let $a \in \Sigma_c^\perp$ and $w \in \Sigma_c^*$. We say that a is *good* for w if $w \in \mathit{Infix}(L_c)$ and either $w = \varepsilon$ or $w = u.a$ for some $u \in \Sigma_c^*$. Intuitively, it may be possible to obtain contents w in channel c when a is the last message sent (no message was sent yet through c if $a = \perp$). Note that the set of words $w \in \Sigma_c^*$ such that a is good for w is a regular language. Moreover, with $\mathbf{w} \in \prod_{c\in\mathit{Ch}} \Sigma_c^*$, we associate the finite set $G(\mathbf{w}) \subseteq \prod_{c\in\mathit{Ch}} \Sigma_c^\perp$ of tuples $\mathbf{a} = (\mathbf{a}_c)_{c\in\mathit{Ch}}$ such that, for all $c \in \mathit{Ch}$, $\mathbf{a}_c$ is good for $\mathbf{w}_c$.

Let $\mathbf{v} \in \mathbb{N}^{\mathit{Cnt}}$ and $\mathbf{a} \in \prod_{c\in\mathit{Ch}} \Sigma_c^\perp$. Abusing notation, we will associate with $\mathbf{v}$ and $\mathbf{a}$ the channel contents $[\![\mathbf{v}]\!]_{\mathbf{a}} \in \prod_{c\in\mathit{Ch}} \Sigma_c^*$ (if it exists). We let $[\![\mathbf{v}]\!]_{\mathbf{a}} = \mathbf{w}$ if $\langle\!\langle \mathbf{w} \rangle\!\rangle = \mathbf{v}$ and $\mathbf{a} \in G(\mathbf{w})$. There is at most one such $\mathbf{w}$ so that this is well-defined. Note that $\langle\!\langle [\![\mathbf{v}]\!]_{\mathbf{a}} \rangle\!\rangle = \mathbf{v}$.

▶ **Example 18.** If we have one channel c and our bounded language is $L_c = (a_1a_2a_3)^*(a_4)^*$, then $[\![(4,0)]\!]_{a_2} = a_2a_3a_1a_2$ and $[\![(2,1)]\!]_{a_4} = a_2a_3a_4$, whereas $[\![(3,1)]\!]_{a_3}$ is undefined. Moreover, $G(a_2a_3) = \{a_3\}$ and $G(\varepsilon) = \{a_1, a_2, a_3, a_4, \perp\}$. ⌟

Given $\mathbf{v}$ and $\mathbf{a}$, we can easily compute $[\![\mathbf{v}]\!]_{\mathbf{a}}$ since there are only finitely many words $\mathbf{w}$ for a given $\mathbf{v}$ such that $\langle\!\langle \mathbf{w} \rangle\!\rangle = \mathbf{v}$. Furthermore, we can also compute $G(\mathbf{w})$ for a given $\mathbf{w}$ as we have finitely many possibilities of $\mathbf{a}$.

Finally, for $\mathbf{a} \in \prod_{c\in\mathit{Ch}} \Sigma_c^\perp$, we let $L_{\mathbf{a}}^{\mathsf{last}} \subseteq A_M^*$ be the set of words σ such that, for all $c \in \mathit{Ch}$, $\mathbf{a}_c$ is the last message sent to c in σ (no message was sent if $\mathbf{a}_c = \perp$). We are now ready to state that runs in the FIFO machine are faithfully simulated by runs in the counter machine (the proof is by induction on the length of the trace):

▶ **Proposition 19.** *Let $\sigma \in A_M^*$. For all $(q, \mathbf{w}) \in S_M$ and $\mathbf{a} \in \prod_{c \in Ch} \Sigma_c^\perp$ such that $\sigma \in L_{\mathbf{a}}^{\mathsf{last}}$, we have: $(q_0, \varepsilon) \xrightarrow{\sigma}_M (q, \mathbf{w}) \implies \big((q_0, \mathbf{0}) \xrightarrow{\langle\!\langle \sigma \rangle\!\rangle}_{\mathcal{C}} (q, \langle\!\langle \mathbf{w} \rangle\!\rangle)$ and $\mathbf{a} \in G(\mathbf{w})\big)$.*

Conversely, we can show that runs of the counter machine can be retrieved in the FIFO machine (again, the proof proceeds by induction on the length of the trace):

▶ **Proposition 20.** *Let $\tau \in A_{\mathcal{C}}^*$. For all $(q, \mathbf{v}) \in S_{\mathcal{C}}$ and $\mathbf{a} \in \prod_{c \in Ch} \Sigma_c^\perp$ such that $\tau \in \langle\!\langle Pref(\mathcal{V}) \cap L_{\mathbf{a}}^{\mathsf{last}} \rangle\!\rangle$, we have: $(q_0, \mathbf{0}) \xrightarrow{\tau}_{\mathcal{C}} (q, \mathbf{v}) \implies (q_0, \varepsilon) \xrightarrow{[\![\tau]\!]}_M (q, [\![\mathbf{v}]\!]_{\mathbf{a}})$.*

From Propositions 19 and 20 and Lemma 17, we obtain the following corollary.

▶ **Corollary 21.** *For all $(q, \mathbf{w}) \in S_M$, we have: $(q, \mathbf{w}) \in Reach_M(\mathscr{L}_!) \iff (q, \langle\!\langle \mathbf{w} \rangle\!\rangle) \in Reach_{\mathcal{C}}(L_{\mathcal{C}}^{\mathsf{zero}} \cap \langle\!\langle \mathcal{V} \cap \bigcup_{\mathbf{a} \in G(\mathbf{w})} L_{\mathbf{a}}^{\mathsf{last}} \rangle\!\rangle)$.*

From Theorem 6, we know that verifying whether $(q, \langle\!\langle \mathbf{w} \rangle\!\rangle) \in Reach_{\mathcal{C}}(L_{\mathcal{C}}^{\mathsf{zero}} \cap L)$ where $L = \langle\!\langle \mathcal{V} \cap \bigcup_{\mathbf{a} \in G(\mathbf{w})} L_{\mathbf{a}}^{\mathsf{last}} \rangle\!\rangle$ is decidable. Hence, we can already deduce decidability of the (configuration-)reachability problem. In fact, using Propositions 19 and 20, we can solve the more general IB rational-reachability problem. For this, it is actually enough to check, in the counter machine, the reachability of a counter value that belongs to a semi-linear set. For $\mathbf{a} \in \prod_{c \in Ch} \Sigma_c^\perp$ and a rational relation $\mathscr{R} \subseteq \prod_{c \in Ch} \Sigma_c^*$, let $V_{\mathbf{a}}(\mathscr{R}) = \{\mathbf{v} \in \mathbb{N}^{Cnt} \mid [\![\mathbf{v}]\!]_{\mathbf{a}} \in \mathscr{R}\}$.

▶ **Lemma 22.** *The set $V_{\mathbf{a}}(\mathscr{R})$ is effectively semi-linear.*

Using this property, we finally reduce the IB rational-reachability problem to a reachability problem in counter machines:

▶ **Corollary 23.** *For every $q \in Q$, we have: $(q, \mathbf{w}) \in Reach_M(\mathscr{L}_!)$ for some $\mathbf{w} \in \mathscr{R} \iff (q, \mathbf{v}) \in Reach_{\mathcal{C}}(L_{\mathcal{C}}^{\mathsf{zero}} \cap \langle\!\langle \mathcal{V} \cap L_{\mathbf{a}}^{\mathsf{last}} \rangle\!\rangle)$ for some $\mathbf{a} \in \prod_{c \in Ch} \Sigma_c^\perp$ and $\mathbf{v} \in V_{\mathbf{a}}(\mathscr{R})$.*

By Theorem 6, we can now deduce Theorem 11, i.e., decidability of IB rational-reachability.

4 Reachability, Deadlock, Unboundedness, and Termination

We now address some other commonly studied reachability problems, which, as it turns out, can be reduced to the IB rational-reachability problem studied in the previous section.

A configuration $(q, \mathbf{w})$ of a FIFO machine M is a *deadlock* if there is no $(q', \mathbf{w}')$ such that $(q, \mathbf{w}) \to_M (q', \mathbf{w}')$.

▶ **Definition 24** (IB decision problems). *Given a FIFO machine $M = (Q, Ch, \Sigma, T, q_0)$, a control-state $q \in Q$, a configuration $s \in S_M$, and a tuple $\mathscr{L} = (L_c)_{c \in Ch}$ of non-empty regular bounded languages $L_c \subseteq \Sigma_c^*$.*

- IB reachability*: Do we have $s \in Reach_M(\mathscr{L}_!)$?*
- IB control-state reachability*: Do we have $(q, \mathbf{w}) \in Reach_M(\mathscr{L}_!)$ for some $\mathbf{w}$?*
- IB deadlock*: Does $Reach_M(\mathscr{L}_!)$ contain a deadlock ?*
- IB unboundedness*: Is $Reach_M(Pref(\mathscr{L}_!))$ infinite?*
- IB termination*: Is there no infinite execution of the form $init_M \xrightarrow{\beta_1} s_1 \xrightarrow{\beta_2} s_2 \xrightarrow{\beta_3} \dots$ such that, for all $i \in \mathbb{N}$, we have $s_i \in S_M$, $\beta_i \in A_M$, and $\beta_1 \dots \beta_i \in Pref(\mathscr{L}_!)$?*

Reachability and Deadlock

In [18], it was shown that reachability reduces to control-state reachability for flat FIFO machines but the converse is not true. However, using the same reductions as in [18], we obtain the following results:

▶ **Proposition 25.** *IB reachability is*
(a) *recursively equivalent to IB control-state reachability for FIFO machines, and*
(b) *recursively reducible to IB deadlock for FIFO machines.*

If, for a given $q \in Q$, we set $\mathscr{R}$ to be the universal relation $\prod_{c \in Ch} \Sigma_c^*$, IB rational-reachability captures IB control-state reachability, and if we set $\mathscr{R} = \{\mathbf{w}\}$, we can decide if the configuration $(q, \mathbf{w})$ is reachable. In order to reduce IB deadlock to IB rational-reachability, we first explore the control states in order to find the set of states $Q' \subseteq Q$ which allow only receptions (no control states with sends can be part of a deadlock). This set can easily be computed from the set of transitions of the machine. Then, for each $q \in Q'$, we can see if there exists a reachable configuration $(q, \mathbf{w})$ such that, for all c, we have $\mathbf{w}_c \in K_c = \{\varepsilon\} \cup \{a.u \mid u \in \Sigma_c^* \text{ and } a \in \Sigma_c \text{ such that there is no transition } (q, \langle c?a \rangle, q') \text{ in } M\}$. Note that $\mathscr{R}_q = \prod_{c \in Ch} K_c$ is recognizable and, therefore, rational. Furthermore, if there exists such a reachable $(q, \mathbf{w})$ with $\mathbf{w} \in \mathscr{R}_q$, then it is a deadlock. Hence, using the fact that generalized IB rational-reachability is decidable (Theorem 11), we immediately deduce the following corollary:

▶ **Corollary 26.** *The problems IB reachability, IB control-state reachability, and IB deadlock are decidable for FIFO machines.*

▶ Remark. Since input-bounded FIFO machines subsume VASS (a VASS can be seen as an input-bounded FIFO machine with an alphabet reduced to a unique letter), the complexity of IB reachability is not elementary, which is inherited from the lower bound for VASS [11].

Unboundedness and Termination

IB unboundedness in FIFO machines reduces to an equivalent problem in counter machines. Given a FIFO machine $\hat{M}$ and $\hat{\mathscr{L}}$, the associated FIFO machine M in normal form (with the corresponding tuple $\mathscr{L}$ of distinct-letter languages), as well as the associated counter machine $\mathcal{C}$, the following result can be derived.

▶ **Proposition 27.** *$Reach_{\hat{M}}(\hat{\mathscr{L}}_!)$ is infinite iff $Reach_M(\mathscr{L}_!)$ is infinite iff $Reach_{\mathcal{C}}(L_{\mathcal{C}}^{\mathsf{zero}} \cap \langle\!\langle \mathcal{V} \rangle\!\rangle)$ is infinite.*

This statement also applies to prefix-closed languages so we have $Reach_{\hat{M}}(Pref(\hat{\mathscr{L}}_!))$, $Reach_M(Pref(\mathscr{L}_!))$, and $Reach_{\mathcal{C}}(L_{\mathcal{C}}^{\mathsf{zero}} \cap \langle\!\langle Pref(\mathcal{V}) \rangle\!\rangle)$ are either all infinite or all finite. The latter-most is decidable as we establish in the following. Recall that, by the construction of $\mathcal{C}$, we have $Reach_{\mathcal{C}} = Reach_{\mathcal{C}}(L_{\mathcal{C}}^{\mathsf{zero}} \cap \langle\!\langle Pref(\mathcal{V}) \rangle\!\rangle)$.

The main idea that follows is the reduction of unboundedness of the counter machine to reachability in a modified counter machine. It is not immediate that we can use the results in the literature (for example [13]) which reduce boundedness to reachability in Petri nets/VASS. This is because the property of monotonicity does not extend to zero tests; if one can execute a zero test at $(q, \mathbf{v})$, it is not necessarily the case that it can be executed at $(q, \mathbf{v}')$ with $\mathbf{v} \leq \mathbf{v}'$, where we let $\mathbf{v} \leq \mathbf{v}'$ if $\mathbf{v}_x \leq \mathbf{v}'_x$ for all $x \in Cnt$. However, we show that, for the counter machine that we construct from the FIFO machine, this property does hold. If we are able to show this, the constructions used in the case of VASS can be adapted.

▶ **Lemma 28.** *For every execution in $\mathcal{C}$ of the form $(q_0, \mathbf{0}) \xrightarrow{\sigma} (q, \mathbf{v}) \xrightarrow{\sigma'} (q, \mathbf{v}')$ such that $\mathbf{v} \leq \mathbf{v}'$, the following holds: The only counters that are tested to zero during σ' already evaluate to zero at $(q, \mathbf{v})$, and do not change their value throughout the execution of σ'.*

Proof. Let us assume to the contrary that there is at least one counter which is incremented or decremented during σ' and also tested to zero during the execution. Without loss of generality, let us consider $x_{(c,i)}$ to be the first counter along the execution σ' that is tested to zero during σ' and also incremented/decremented before it was tested to zero.

- Case (1): It has a non-zero value at $(q, \mathbf{v})$, and is then either decremented, or first incremented and then decremented, and finally tested to zero. Since we know that $\sigma.\sigma' \in L_{\mathcal{C}}^{\mathsf{zero}}$, no counter tested to zero can then be incremented. Hence, its value will remain zero. But this is a contradiction to our assumption that $\mathbf{v} \leq \mathbf{v}'$. Hence, all the counters with non-zero values at $(q, \mathbf{v})$ cannot be tested to zero during σ'.
- Case (2): It has value zero in $(q, \mathbf{v})$, and is incremented, then decremented, then tested to zero during σ'. This implies that it first has to be incremented. Consider now some sub-execution $\sigma'' \in \mathit{Pref}(\sigma')$ where $(q, \mathbf{v}) \xrightarrow{\sigma''} (q_1, \mathbf{v}_1)$ such that the value of $x_{(c,i)}$ in the configuration $(q_1, \mathbf{v}_1)$ is non-zero. Since there are no "new" zero-tests along the execution σ'' (by our assumption), we can execute σ'' from $(q, \mathbf{v}')$ (by the monotonicity and trace property). However, we cannot increment the counter $x_{(c,i)}$ along σ'', because it was tested to zero during the run $(q_0, \mathbf{0}) \xrightarrow{\sigma.\sigma'} (q, \mathbf{v}')$. Hence, we once again have a contradiction. ◀

Now, we can use results from [19] to show the following:

▶ **Proposition 29.** *The set $\mathit{Reach}_{\mathcal{C}}$ is infinite iff there exist $\sigma, \sigma' \in A_{\mathcal{C}}^*$, $q \in Q$, and $\mathbf{v}, \mathbf{v}' \in \mathbb{N}^{Cnt}$ such that $(q_0, \mathbf{0}) \xrightarrow{\sigma} (q, \mathbf{v}) \xrightarrow{\sigma'} (q, \mathbf{v}')$ and $\mathbf{v} < \mathbf{v}'$ (i.e., $\mathbf{v} \leq \mathbf{v}'$ and $\mathbf{v} \neq \mathbf{v}'$).*

Construction of modified counter machine. We modify the counter machine $\mathcal{C}$ and construct a new counter machine $\mathcal{C}'$ such that $\mathit{Reach}_{\mathcal{C}}$ is infinite iff a configuration belonging to a finite set is reachable in $\mathcal{C}'$. The construction is loosely based on the reduction of boundedness to reachability for Petri Nets in [13]. Since we do not know the values of $\mathbf{v}$ and $\mathbf{v}'$ a priori, we will try to characterize the general condition. The difference $\mathbf{v}' - \mathbf{v}$ is a non negative vector, with at least one strictly positive component. We add a duplicate set of counters for every counter in the system. The intuition is that the counter machine non-deterministically moves from operating on both sets to a configuration from where it only operates on this second set. The first set will remain unchanged (with the value $\mathbf{v}$), and the second set will keep track of the values (until it reaches $\mathbf{v}'$). From this configuration (which represents $(q, \mathbf{v}')$), we move to a new control state, q_{reach}. Here, we check for the condition $\mathbf{v}' - \mathbf{v} > \mathbf{0}$ by first decrementing each counter in the first set which has a non-zero value in tandem with the corresponding counter in the second set. We do this until all the counters in the first set are equal to zero. If $\mathbf{v}' - \mathbf{v} > \mathbf{0}$, then there is at least one counter in the second set with a non-zero counter value. We non-deterministically decrement all the counters in the second set until we reach a configuration that has some counter c in the second set with a value of 1, and all other counters evaluate to zero. Since there are finitely many such configurations, we can just check every case.

Note that we can extend all these results for the case of termination as well. The only difference is that we now consider configurations $(q, \mathbf{v})$ and $(q, \mathbf{v}')$ such that $\mathbf{v} \leq \mathbf{v}'$. Once again, we can follow a similar argument to reduce the termination to the reachability of a configuration in this same modified counter machine.

Hence, we obtain the following theorem:

▶ **Theorem 30.** *IB unboundedness and IB termination are decidable for FIFO machines.*

▶ Remark. Gouda et al, stated that unboundedness is in EXPSPACE for letter-bounded systems [23]. However, they only give an idea of the proof, stating that it can be done in a similar fashion as for the deadlock problem. In the construction for solving the deadlock problem, they reduce the input language to *tally* letter-bounded languages (tally means that the input-language is included in a^* where a is a letter). They add as many channels as letters in the original letter-bounded-language. Furthermore, in order to ensure that every channel is empty before the next channel is read, they ensure that in all control states where a later channel is being read, there are reception transitions of previous channel contents which lead to a sink state (where there is never a deadlock). Notice that it is still possible to leave a channel non-empty before the next channel is read. But one never reaches a deadlock in such an "incorrect" run, since there is always the option of reading the unread channel contents of the previous channels and reach the sink state.

However, when we consider this model for unboundedness, there may exist unbounded "incorrect" runs since we can leave a channel non-empty and proceed to the next and may have an unbounded run. Hence, it seems that one still needs some reachability test to check if the runs are correct as we cannot ensure that some channels are zero in an unbounded run.

5 Conclusion and Perspectives

We extend recent results of the *bounded verification* of communicating finite-state machines (equivalently FIFO machines) [14] and of *flat* FIFO machines [18] by using bounded languages for controlling the input-languages of FIFO channels (and not for controlling the runs of the machine). We extend old and recent results about input-bounded FIFO machines (see Table 1). In particular, we introduce the rational-reachability problem, which subsumes most of the well-known variants of reachability problems like: the (classical) reachability problem, the control-state reachability problem, and the deadlock problem. We also unify the terminology to facilitate the comparison between results. Moreover, note that, for most problems (except general/rational reachability), we can reduce *output-bounded* reachability to an equivalent input-bounded problem. There are still many open problems and challenges:

- What is the precise complexity of the five problems for input-bounded FIFO machines with a fixed number of channels?
- What is the precise complexity of control-state reachability, deadlock, unboundedness, and termination for input-bounded FIFO machines?
- The size of the counter machine associated with a FIFO machine and a tuple of bounded languages is exponential, but only polynomial when we start from a normal form. It will be interesting to see whether the use of existing tools for counter machines is feasible for the verification of FIFO machines from case studies. Case studies shall also reveal how many FIFO machines/systems are actually boundable and/or flattable.

Towards a theory of boundable FIFO machines. In Example 9, we have seen that all configurations that are reachable in the CDP protocol are already reachable in presence of a suitable collection $\mathscr{L}$ of bounded input-languages. By analogy with the well-established theory of *flattable* machines [3, 12, 8], we propose the following definition.

Table 1 Summary of key results; results for all other extensions are subsumed by these results (D stands for decidable).

	Flat	Letter-bounded	Bounded
UNBOUND	NP-C ([18])	D ([23])	D ([26])
TERM	NP-C ([18])	D	**D**
REACH	NP-C ([18])	D	**D, not ELEM**
CS-REACH	NP-C ([14, 18])	D	D
DEADLOCK	D	D ([23])	**D**

▶ **Definition 31.** *Let M be a FIFO machine and let $\mathscr{L}$ be a tuple of regular bounded languages. We say that M is* $\mathscr{L}$-boundable *if $Reach_M = Reach_M(\mathscr{L}_!)$. We say that M is* boundable *if there exists a tuple $\mathscr{L}$ of regular bounded languages such that M is* $\mathscr{L}$-boundable.

Hence, we deduce that reachability is decidable for $\mathscr{L}$-boundable FIFO machines, which is a *strictly larger* class than input-bounded machines. CDP is not input-bounded but it is $\mathscr{L}_{CDP}$-boundable with $\mathscr{L}_{CDP} = ((ab)^*(a+\varepsilon)(ab)^*, e^*)$. Let us also remark that CDP is flattable by using the bounded set of runs $(!a!b)^*!a!e?e(!a!b)^* + (!a!b)^*$ (where we omit channel information for readability), because it covers the reachability set which is equal to $(ab)^*(a+\varepsilon)(ab)^*$ on control-state $(0,0)$. It is not clear whether reachability is decidable for boundable machines. A strategy that would fairly enumerate *all* regular bounded families $\mathscr{L}_1, \mathscr{L}_2, \ldots, \mathscr{L}_n, \ldots$ will necessarily find the good one, if M is boundable, but this is not sufficient because we must be able to *recognize* $Reach_M$. Observe that boundable machines are more robust than flat machines. Consider a system $\mathcal{S} = (\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n)$ of n flat finite automata $\mathcal{A}_i$ communicating peer to peer (P2P) through one-directional FIFO channels. Let $M_{\mathcal{S}}$ denote FIFO the machine obtained as the Cartesian product of all automata $\mathcal{A}_i$ of $\mathcal{S}$; there is no reason to assume that $M_{\mathcal{S}}$ is flattable but it is input-bounded and thus $M_{\mathcal{S}}$ is $\mathscr{L}$-boundable where $\mathscr{L}$ is computable from $\mathcal{S}$.

References

1. Parosh Aziz Abdulla, Aurore Collomb-Annichini, Ahmed Bouajjani, and Bengt Jonsson. Using forward reachability analysis for verification of lossy channel systems. *Formal Methods Syst. Des.*, 25(1):39–65, 2004. doi:10.1023/B:FORM.0000033962.51898.1a.
2. C. Aiswarya, Paul Gastin, and K. Narayan Kumar. Verifying communicating multi-pushdown systems via split-width. In *Automated Technology for Verification and Analysis - 12th International Symposium, ATVA 2014*, volume 8837 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2014.
3. Sébastien Bardin, Alain Finkel, Jérôme Leroux, and Laure Petrucci. FAST: Acceleration from theory to practice. *International Journal on Software Tools for Technology Transfer*, 10(5):401–424, October 2008. doi:10.1007/s10009-008-0064-3.
4. Jean Berstel. *Transductions and context-free languages*, volume 38 of *Teubner Studienbücher : Informatik*. Teubner, 1979.
5. Ahmed Bouajjani, Constantin Enea, Kailiang Ji, and Shaz Qadeer. On the completeness of verifying message passing programs under bounded asynchrony. In Hana Chockler and Georg Weissenbacher, editors, *Computer Aided Verification - 30th International Conference, CAV 2018, Held as Part of the Federated Logic Conference, FloC 2018, Proceedings, Part II*, volume 10982 of *Lecture Notes in Computer Science*, pages 372–391. Springer, 2018. doi:10.1007/978-3-319-96142-2_23.
6. Daniel Brand and Pitro Zafiropulo. On communicating finite-state machines. *J. ACM*, 30(2):323–342, 1983. doi:10.1145/322374.322380.

7 Gérard Cécé and Alain Finkel. Verification of programs with half-duplex communication. *Inf. Comput.*, 202(2):166–190, 2005. doi:10.1016/j.ic.2005.05.006.

8 Pierre Chambart, Alain Finkel, and Sylvain Schmitz. Forward analysis and model checking for trace bounded WSTS. In Lars M. Kristensen and Laure Petrucci, editors, *Proceedings of the 32nd International Conference on Applications and Theory of Petri Nets (PETRI NETS'11)*, volume 6709 of *Lecture Notes in Computer Science*. Springer, 2011. doi:10.1007/978-3-642-21834-7_4.

9 Pierre Chambart and Philippe Schnoebelen. The ordinal recursive complexity of lossy channel systems. In *Proceedings of the Twenty-Third Annual IEEE Symposium on Logic in Computer Science, LICS 2008*, pages 205–216. IEEE Computer Society, 2008. doi:10.1109/LICS.2008.47.

10 Christian Choffrut. Relations over words and logic: A chronology. *Bulletin of the EATCS*, 89:159–163, 2006.

11 Wojciech Czerwinski, Slawomir Lasota, Ranko Lazic, Jérôme Leroux, and Filip Mazowiecki. The reachability problem for petri nets is not elementary. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019*, pages 24–33. ACM, 2019.

12 Stéphane Demri, Alain Finkel, Valentin Goranko, and Govert van Drimmelen. Model-checking CTL* over flat Presburger counter systems. *Journal of Applied Non-Classical Logics*, 20(4):313–344, 2010. doi:10.3166/jancl.20.313-344.

13 Catherine Dufourd and Alain Finkel. Polynomial-Time Many-One Reductions for Petri Nets. In S. Ramesh and G. Sivakumar, editors, *Foundations of Software Technology and Theoretical Computer Science, 17th Conference*, volume 1346 of *Lecture Notes in Computer Science*, pages 312–326. Springer, 1997. doi:10.1007/BFb0058039.

14 Javier Esparza, Pierre Ganty, and Rupak Majumdar. A perfect model for bounded verification. In *Proceedings of the 27th Annual IEEE Symposium on Logic in Computer Science, LICS 2012*, pages 285–294. IEEE Computer Society, 2012. doi:10.1109/LICS.2012.39.

15 Alain Finkel. About monogeneous fifo Petri nets. In *Proceedings of the 3rd International Conference on Applications and Theory of Petri Nets (APN'82)*, Varenna, Italy, September 1982.

16 Alain Finkel. Decidability of the termination problem for completely specified protocols. *Distributed Computing*, 7(3):129–135, 1994. doi:10.1007/BF02277857.

17 Alain Finkel and Annie Choquet. Simulation of linear fifo nets by Petri nets having a structured set of terminal markings. In *Proceedings of the 8th International Conference on Applications and Theory of Petri Nets (APN'87)*, Zaragoza, Spain, June 1987.

18 Alain Finkel and M. Praveen. Verification of flat FIFO systems. In Wan Fokkink and Rob van Glabbeek, editors, *30th International Conference on Concurrency Theory, CONCUR 2019*, volume 140 of *LIPIcs*, pages 12:1–12:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.CONCUR.2019.12.

19 Alain Finkel and Philippe Schnoebelen. Well-structured transition systems everywhere! *Theor. Comput. Sci.*, 256(1-2):63–92, 2001. doi:10.1016/S0304-3975(00)00102-X.

20 Blaise Genest, Dietrich Kuske, and Anca Muscholl. On communicating automata with bounded channels. *Fundam. Inform.*, 80(1-3):147–167, 2007. URL: http://content.iospress.com/articles/fundamenta-informaticae/fi80-1-3-09.

21 Seymour Ginsburg and Edwin H. Spanier. Bounded Algol-Like Languages. *Transactions of the American Mathematical Society*, 113(2):333–368, 1964. URL: http://www.jstor.org/stable/1994067.

22 Cinzia Di Giusto, Laetitia Laversa, and Étienne Lozes. On the k-synchronizability of systems. In Jean Goubault-Larrecq and Barbara König, editors, *Foundations of Software Science and Computation Structures - 23rd International Conference, FOSSACS 2020*, volume 12077 of *Lecture Notes in Computer Science*, pages 157–176. Springer, 2020. doi:10.1007/978-3-030-45231-5_9.

23 M. G. Gouda, E. M. Gurari, T. H. Lai, and L. E. Rosier. On deadlock detection in systems of communicating finite state machines. *Comput. Artif. Intell.*, 6(3):209–228, July 1987.

24 Alexander Heußner, Jérôme Leroux, Anca Muscholl, and Grégoire Sutre. Reachability analysis of communicating pushdown systems. In C.-H. Luke Ong, editor, *Foundations of Software Science and Computational Structures, 13th International Conference, FOSSACS 2010*, volume 6014 of *Lecture Notes in Computer Science*, pages 267–281. Springer, 2010. doi:10.1007/978-3-642-12032-9_19.

25 Thierry Jéron. Testing for unboundedness of FIFO channels. In Christian Choffrut and Matthias Jantzen, editors, *STACS 91, 8th Annual Symposium on Theoretical Aspects of Computer Science*, volume 480 of *Lecture Notes in Computer Science*, pages 322–333. Springer, 1991. doi:10.1007/BFb0020809.

26 Thierry Jéron and Claude Jard. Testing for unboundedness of FIFO channels. *Theor. Comput. Sci.*, 113(1):93–117, 1993. doi:10.1016/0304-3975(93)90212-C.

27 Salvatore La Torre, P. Madhusudan, and Gennaro Parlato. Context-bounded analysis of concurrent queue systems. In C. R. Ramakrishnan and Jakob Rehof, editors, *Tools and Algorithms for the Construction and Analysis of Systems, 14th International Conference, TACAS 2008*, volume 4963 of *Lecture Notes in Computer Science*, pages 299–314. Springer, 2008. doi:10.1007/978-3-540-78800-3_21.

28 P. Madhusudan and Gennaro Parlato. The tree width of auxiliary storage. In *Proceedings of the 38th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2011*, pages 283–294. ACM, 2011.

29 Ernst W. Mayr. An algorithm for the general petri net reachability problem. *SIAM J. Comput.*, 13(3):441–460, 1984.

30 Gérard Memmi and Alain Finkel. An introduction to fifo nets-monogeneous nets: A subclass of fifo nets. *Theor. Comput. Sci.*, 35:191–214, 1985. doi:10.1016/0304-3975(85)90014-3.

31 Bernard Vauquelin and Paul Franchi-Zannettacci. Automates a file. *Theor. Comput. Sci.*, 11:221–225, 1980. doi:10.1016/0304-3975(80)90047-X.

32 Yao-Tin Yu and Mohamed G. Gouda. Unboundedness detection for a class of communicating finite-state machines. *Inf. Process. Lett.*, 17(5):235–240, 1983. doi:10.1016/0020-0190(83)90105-9.